UMass Boston CS 410 f25 Homework 3
Posted Sunday, 7 December, 2025
Due Friday, 12 December, 2025 at 11:59 pm

J.H.DeBlois

0 Directions

Homework must be typed and converted to Portable Document Format (PDF). Drawings can be hand-written, photographed and included. To submit your homework, prepare one PDF file called `hw2.pdf` — the filename must be exactly `hw2.pdf`, otherwise it will not be collected. Upload the file to your course directory, the `cs410` folder linked to your home directory on the CS Linux server. Please do not put your file in a subdirectory.
If you have trouble with uploading, email `operator@cs.umb.edu` for help and copy me.
The questions in this homework are based on the reading in Essential Scrum (ES) by Kenneth S. Rubin, 2013. For this homework, the reading is:

- ES, Chapter 22 Sprint Retrospective pp375-394,
- ES, Chapter 23 The Path Forward pp395-399 and
- ES, T-Shaped Skills, Figure 11-6, p202, and Chapter 11 Development Team, Characteristics/Skills, Self-organizing, pp198-200 and T-Shaped Skills, pp201-203.
- ES, Chapter 20 Sprint Execution, Flow management pp349-350.

You may ask for an extension if you need it.

1 Sprint Retrospective

We all start new things with a vision. This is true for a Sprint Retrospective.

1.1 Your Client's Vision

For the long project, each of our clients wrote a statement of what is to be built. The statements are posted on the class website. Please reread the one for your team.
In general, the clients are very satisfied with the work of each of our teams. So in choosing a focus for the retrospective this does not have to be our focus.

1.2 Set the confidence threshold and do a knowledge acquisition sprint

Rubin gave an illustration of how a knowledge-acquisition sprint could achieve the confidence threshold selected. Each team has done some knowledge acquisition tasks.
In general, all three teams performed their knowledge acquisition adequately. In some cases, outside software was chosen without much communication of the specifics. It's good to have a list of the options to choose from before you choose.
Often the client will need to make a change, so the more important thing is to understand how the outside software your team has chosen works.

1.3 The overall plan

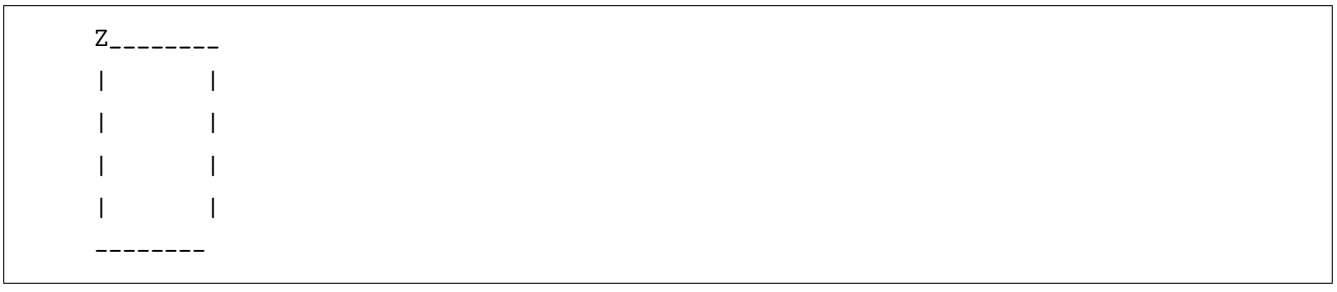
Each team likely did this for each sprint:

- Select a goal for how to improve the current product.
- List the requirements for what you plan. (Then make cards for requirements you will build to - Make them 3-part statements showing value. You might have 5-7. Some could say look up information - knowledge acquisition.)
- Design what you plan to build. (What will it do different? How will you test each part of it? You should get a list of tests AHEAD of your building any of it. We let the test drive the design.
- Then build it.
- When it is done, show it by running it, not by talking about it. (When it runs, it should look different from last time. Play with it, to see what added value to the system and what you think still could be better. Then look at the code. It should be carefully commented with who wrote it, what feature is where and dates.
- There is a long chain of processing going on now for each of you as your run your teams's software. Why not start by drawing a diagram of what you know about that chain of steps. It's a system diagram, which everyone can use. It's explained in the next paragraph.

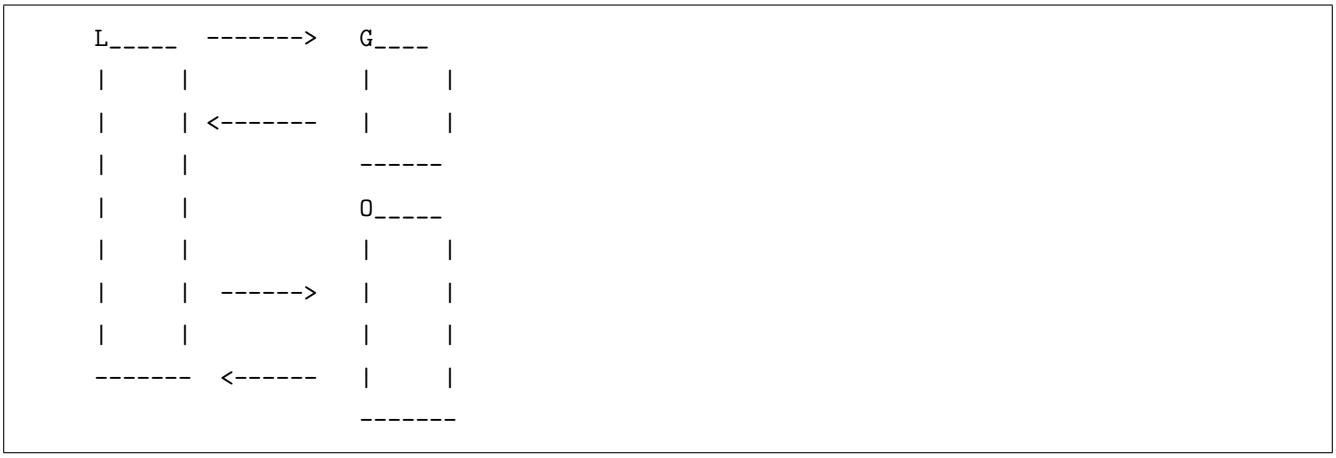
2 Draw a System Diagram

(50 points) The first goal is to get all parts of your team's software into your system diagram. You probably know some parts better than others. It's okay to guess. You want the whole diagram to connect the parts.
You are welcome to draw your system diagram by hand, photograph it and include it in your .pdf.

- Draw a box and lable it with your team short name. Put the name outside the box. I'm using Z.



- Take the real estate view - always the best way to figure out software. Computers are physical objects so they have location and are owned. When you operate software you are at a console. There could be several other locations with computers involved.
- Let's use L for your laptop or desktop.
- Let's use G for our access to Github.
- Let's use O, P, Q, etc. for Other, etc.
- Put these inside your big box.



- Then make some smaller boxes to go inside each of the boxes you have drawn.
- Keep going until you have all the parts. No need to be fancy. Just go along like this document in a .tex file, if you use LaTeX. Or use a graphics program.
- You don't actually have to try to fit all the boxes inside each other. I don't want you to have to keep redrawing. It is fine to lable the from box and show the to box. Just be sure to note which outer box you are in before you start a new box.

3 The Path Forward

Discovering your own path is almost the most important thing. Learn from your teammates, but recognize what you like to do and can do.
Sharing best practices helps you develop your skills for the longer term.

3.1 Your Path Forward

(20 points) Please write a paragraph about discovering your own path as you did your team work this semester.

3.2 Best Practices

Please write a second paragraph about best practices longterm. Think about what have you seen work well for your team.
(30 points) List three items you recommend. Start by writing a sentence on each one. Indicate when they each became clear to you. Then select one to elaborate on. Try to reflect what you learned about agile methods. Try to also be practical. Then write your recommendation for your team best practices longterm - things to focus on better.